

A CRM Application to Handle the Clients and their property Related Requirements

- A Surya Daniel

Project Overview

Dreams World Properties integrates Salesforce to streamline customer interactions. Website engagement triggers automated record creation in Salesforce, capturing customer details and preferences. Salesforce categorizes users as approved or non-approved, offering tailored property selections to approved users. This enhances user experience and efficiency, providing personalized recommendations and broader listings. Seamless integration optimizes operations, improving customer engagement and facilitating growth in the real estate market.

Objectives

Business Goals:

- **Improve Client Management:** Maintain a comprehensive record of client information, preferences, and interactions.
- **Enhance Customer Satisfaction:** Provide personalized service by quickly addressing client property requirements.
- **Streamline Property Management:** Efficiently handle property listings, viewings, and transactions.
- **Boost Sales Efficiency:** Enable sales teams to track and manage leads and opportunities more effectively.
- **Data-Driven Insights:** Generate reports and insights to understand market trends and client needs.

Specific Outcomes:

- **Faster Client Response Times:** Reduced response times to client inquiries.
- **Increased Conversion Rates:** More effective lead tracking and follow-ups.
- **Improved Customer Retention:** Enhanced relationships with personalized interactions.
- **Increased Productivity:** Sales and service teams spend less time on manual data entry and more on client engagement.
- **Better Market Understanding:** Enhanced analytics for market trends and demand prediction.

Salesforce Key Features and Concepts Utilized:

- **Sales Cloud:** Manage leads, opportunities, and accounts.
- **Service Cloud:** Track customer inquiries and provide support.
- **Property Management Objects:** Custom objects for properties, listings, and client requirements.
- **Reports and Dashboards:** Visualize key metrics for property and client performance.
- **Mobile Accessibility:** Manage CRM from any device for on-the-go updates.
- **Automation and Workflow Rules:** Automate notifications and follow-up reminders.

Detailed Steps to Solution Design

Create a Jotform and integrate it with the org to create a record of customers automatically.

- Open your browser and search for jotform and log in.
- After login click on create form and click on start from scratch
- Now create a form to get the customer details like Name, Phone,Email, Address and type of property the customer is interested in.
- Once the form is created, publish it by clicking on publish.

The screenshot displays the Jotform Form Builder interface. At the top, the Jotform logo is on the left, and the 'Form Builder' dropdown menu is next to it. The top navigation bar includes 'BUILD', 'SETTINGS', and 'PUBLISH' tabs, with 'PUBLISH' currently selected. On the right of the top bar, there are buttons for 'Add Collaborators', 'Help', and a user profile icon. Below the top bar, a 'Preview Form' toggle switch is visible. The left sidebar contains several options: 'QUICK SHARE' (Direct form link and social share), 'EMBED' (Various web page embed options), 'ASSIGN FORM' (Assign your forms to others), 'EMAIL' (Reminders and instant sharing), 'PREFILL' (Pre-populate your forms), 'PDF' (Download fillable PDF), and 'PLATFORMS' (Third-party publish options, marked as 'NEW'). The main content area is titled 'DIRECT LINK OF YOUR FORM' and states 'Your form is securely published and ready to use at this address'. It features a 'SHARE WITH LINK' section with a 'Public Form' label and a 'Settings' gear icon. The link displayed is 'https://form.jotform.com/243101034249040', with 'COPY LINK' and 'OPEN IN NEW TAB' buttons. Below this is an 'INVITE BY EMAIL' section with a 'To:' field for entering email addresses. At the bottom, there is a 'SHARE FORM' section with social media icons for Facebook, WhatsApp, X, and LinkedIn, along with a QR code and a '+ View more' link. A 'Give Feedback' button is located in the bottom right corner.

Create Objects from Spreadsheet.

Create Customer Object

- Go to your object manager and click on create object from spreadsheet.
- Click on the link to get the spreadsheet
- customer
- After downloading, upload the file, map the fields and upload to create an object.

The screenshot displays the Salesforce Setup interface, specifically the 'Object Manager' section for the 'Customer' object. The breadcrumb trail at the top reads 'SETUP > OBJECT MANAGER'. The main heading is 'Customer'. On the left, a sidebar lists various configuration options: Details (selected), Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Search Layouts, List View Button Layout, Object Access, Triggers, and Flow Triggers. The 'Details' section is expanded, showing a table of configuration options. The table has two columns: the first column lists the configuration item, and the second column shows the current value. The items listed are: Description (empty), API Name (Customer), Enable Reports (Custom), Singular Label (Customer), Track Activities (Track Field History), Plural Label (Customers), Deployment Status (Help Settings), and a Help Settings link (Standard salesforce.com Help Window).

Details	
Description	
API Name	Customer
Enable Reports	Custom
Singular Label	Customer
Track Activities	Track Field History
Plural Label	Customers
Deployment Status	Help Settings
Help Settings	Standard salesforce.com Help Window

Create Property Object

- Follow the same from the customer object to create the Property Object
- Property

SETUP > OBJECT MANAGER

Property12

Details

Fields & Relationships

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Record Types

Related Lookup Filters

Restriction Rules

Scoping Rules

Object Access

Triggers

Flow Triggers

Details

Description

API Name

Property12__c

Custom

✓

Singular Label

Property12

Plural Label

Properties

Enable Reports

Track Activities

Track Field History

Deployment Status

Deployed

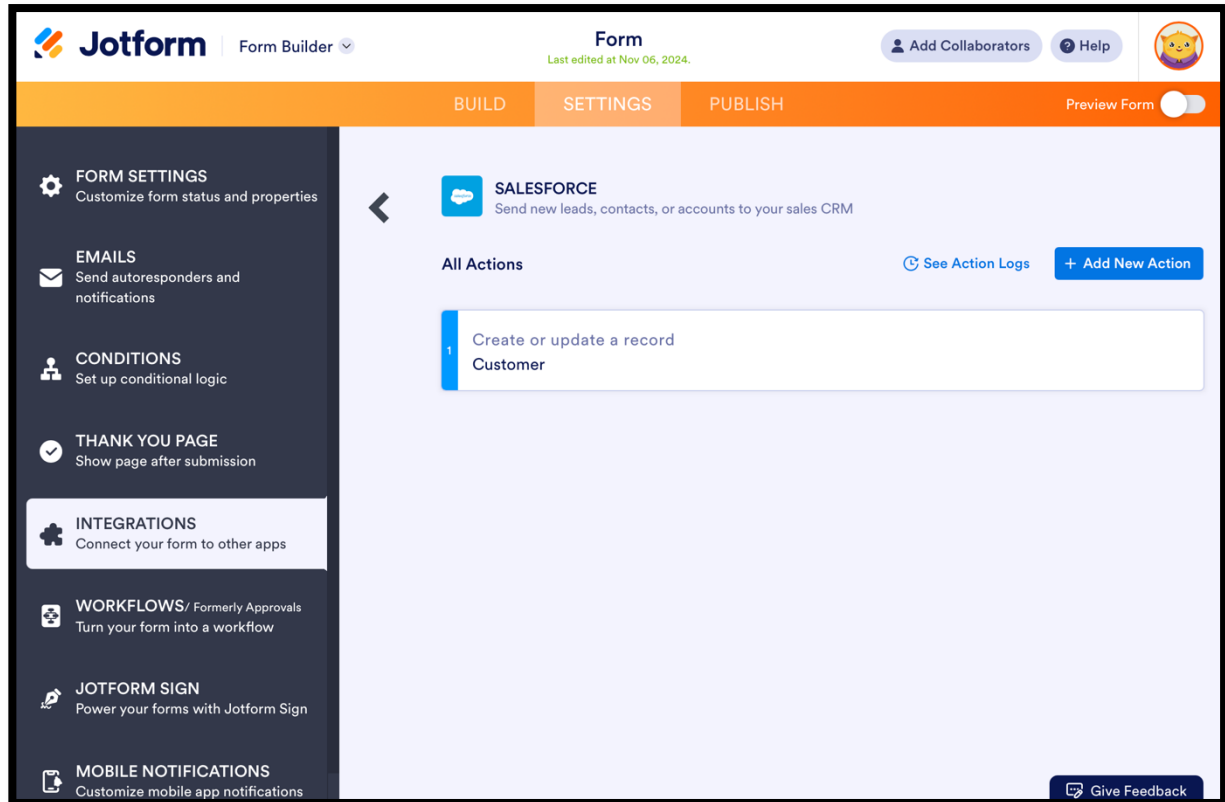
Help Settings

Standard salesforce.com Help Window

EditDelete

Integrate Jotform With Salesforce Platform

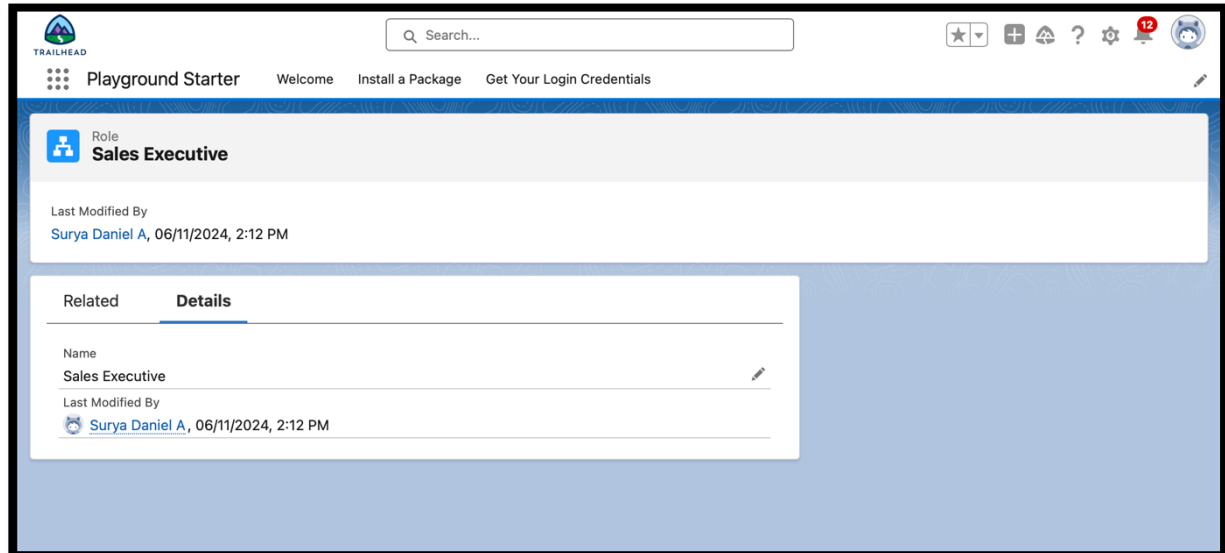
- On the Jotform Platform, Click on Integration and choose Salesforce.
- Click on User Integration and choose “Add to From”.
- Select the Org with which you want to Integrate your jotform with.
- Select an Action - Create a record.
- Select a Salesforce Object : - Customer
- Map Each and every field on the Object with the fields on the form and “Save Action”.
- Then “Save the Integration” and “Finish”.



Create Roles

Sales Executive Role

- Go to Setup and Click on Roles, then click on Expand all and Add a Role just below the Sales Representative
- Label - Sales Executive
- Reports to - Sales Representative
- Similarly Create a Role Name “Sales Manager” below Sales Executive which reports to Sales Executive, Also Add a Role below Sales Manager labeled as “Customer” which reports to Sales Manager.



Create A Property Details App

- From Setup>> Go to App Manager and click on New Lightning App and Name it as “Property Details” and add “Customer” and “Property” Object.
- Click Next >> Next >> Save and Add “System Admin ”Profile.

The screenshot shows the 'App Settings' page in the Lightning App Builder. The left sidebar contains 'App Settings' with sub-items: 'App Details & Branding' (selected), 'App Options', 'Utility Items (Desktop Only)', 'Navigation Items', and 'User Profiles'. The main content area is titled 'App Details & Branding' and includes instructions: 'Give your Lightning app a name and description. Upload an image and choose the highlight color for its navigation bar.'

App Details

- * App Name: Property Details
- * Developer Name: Property_Details
- Description: Enter a description...

App Branding

- Image: Upload button
- Primary Color Hex Value: #0070D2
- Org Theme Options: ☐ Use the app's image and color instead of the org's custom theme
- App Launcher Preview: A button labeled 'PD' with the text 'Property Details' next to it.

Create Profiles

- From Setup >> Go to Profiles and Clone Salesforce Platform User and Name it “Customer”..
- Uncheck all the Custom Objects and Check only Property Details From Custom App Settings.
- Also Remove all the Standard Object Permissions.
- Uncheck all the Custom Object Permissions and check read and view all in “Property”

The screenshot shows the 'Profile Edit' page for the 'Customer' profile. The top banner reads 'It's Better in Lightning' and 'Move to Lightning Experience and give your users a productivity boost.' The left sidebar contains 'Quick Find / Search...', 'Lightning Experience Transition Assistant', 'Salesforce Mobile Quick Start', and a navigation menu with 'Home', 'Administrator', 'Release Updates', 'Manage Users', 'Users', 'Mass Email Users', 'Roles', 'Permission Sets', 'Permission Set Groups', 'User Management Settings', 'Profiles' (selected), 'Public Groups', 'Queues', 'Login History', 'Identity Provider Event Log', and 'Identity Verification History'.

Profile Edit

Name: Customer
User License: Salesforce Platform
Description: [Empty text area]
Custom Profile: ☒

Custom App Settings

	Visible	Default		Visible	Default
Analytics Studio (standard__Insights)	<input type="checkbox"/>	<input type="radio"/>	Playground Starter (trifidips__Playground_Starter)	<input type="checkbox"/>	<input type="radio"/>
App Launcher (standard__AppLauncher)	<input type="checkbox"/>	<input type="radio"/>	Property Details (Property_Details)	<input checked="" type="checkbox"/>	<input checked="" type="radio"/>
Platform (standard__Platform)	<input type="checkbox"/>	<input type="radio"/>	WDC (standard__Work)	<input type="checkbox"/>	<input type="radio"/>

Connected App Access

Ant Migration Tool	<input type="checkbox"/>	Salesforce for Outlook	<input type="checkbox"/>
Dataloader Bulk	<input type="checkbox"/>	Salesforce Mobile Dashboards	<input type="checkbox"/>
Dataloader Partner	<input type="checkbox"/>	Salesforce Touch	<input type="checkbox"/>
Force.com IDE	<input type="checkbox"/>	Workbench	<input type="checkbox"/>

Service Provider Access

Tab Settings

Manager

- From Setup >> Go to Profiles and Clone Salesforce Platform User and Name it “Manager”..
- Uncheck all the Custom Objects and Check only Property Details From Custom App Settings.
- Also Remove all the Standard Object Permissions.
- Uncheck all the Custom Object Permissions and check only “modify all” from “Property” and “Customer”

The screenshot shows the 'Profile Edit' page for a profile named 'Manager'. The page is titled 'It's Better in Lightning' and includes a 'Lightning Experience Transition Assistant' banner. The left sidebar contains navigation links for 'Home', 'Administrator', 'Release Updates', 'Manage Users', 'Profiles', 'Public Groups', 'Queues', 'Login History', 'Identity Provider Event Log', and 'Identity Verification History'. The main content area is divided into several sections: 'Profile Edit' (Name: Manager, User License: Salesforce Platform, Custom Profile: checked), 'Custom App Settings' (a table with columns for Visible and Default for various apps like Analytics Studio, App Launcher, Platform, Playground Starter, Property Details, and WDC), 'Connected App Access' (a table with columns for Visible and Default for various connected apps like Ant Migration Tool, Dataloader Bulk, Dataloader Partner, Force.com IDE, Salesforce for Outlook, Salesforce Mobile Dashboards, Salesforce Touch, and Workbench), 'Service Provider Access', and 'Tab Settings'.

Create A Check Box Field On User

- Setup >> Object Manager >> Search for User >> Fields and Relationships
- Create new Field Named as “Verified” as Data type “Check Box”

The screenshot shows the 'Setup' page in Salesforce, specifically the 'Object Manager' section for the 'User' object. The left sidebar contains navigation links for 'Details', 'Fields & Relationships', 'User Page Layouts', 'User Profile Page Layouts', 'Lightning Record Pages', 'Buttons and Links', 'Compact Layouts', 'Field Sets', 'Object Limits', 'Related Lookup Filters', 'Search Layouts', 'List View Button Layout', 'Triggers', 'Flow Triggers', and 'Validation Rules'. The main content area is titled 'Fields & Relationships' and shows a table with columns for FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. The table contains one row with the field label 'Verified', field name 'Verified__c', and data type 'Checkbox'. There is a search bar with the text 'veri' and buttons for 'New', 'Deleted Fields', and 'Field Dependencies'.

Create Users

User 1

- Go to Setup --> Administration --> Users --> New User
- Last Name - Executive
- Role - Sales Executive
- License - Salesforce
- Profile - System Administrator
- Save

User 2

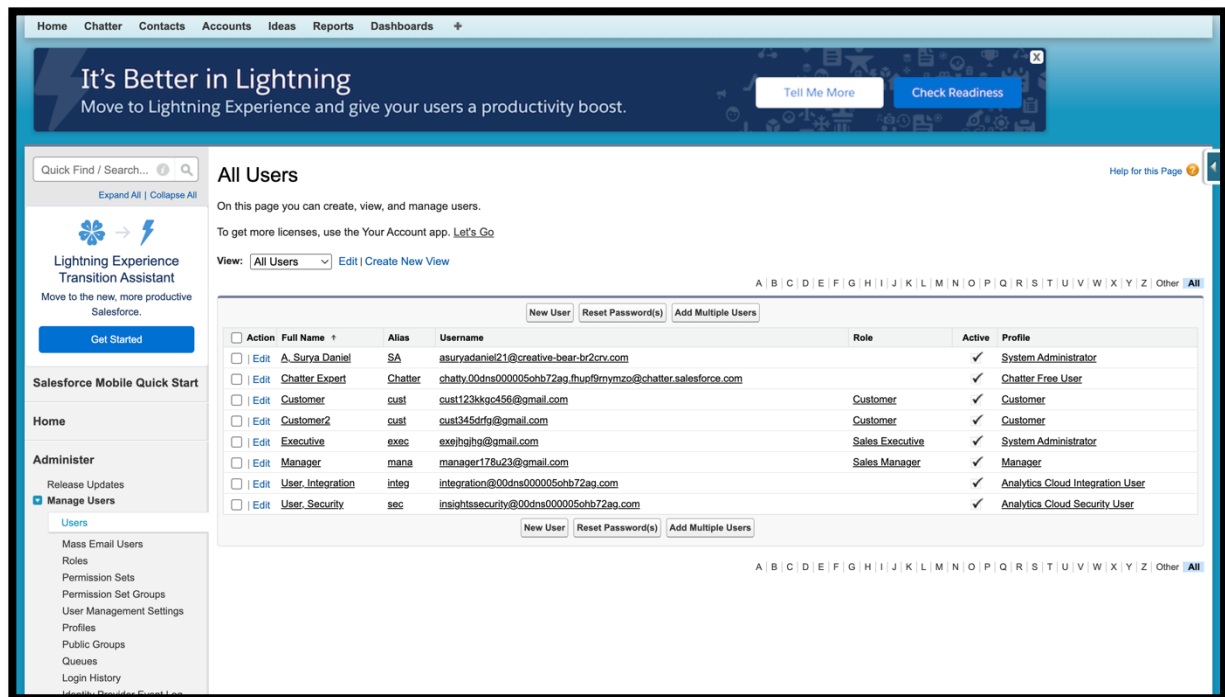
- Go to Setup >> Administration >> Users >> New User
- Last Name >> Manager
- Role >> Sales Manager
- License >> Salesforce Platform
- Profile >> Manager
- Save

User 3

- Go to Setup >> Administration >> Users >> New User
- Last Name >> Customer
- Role >> Customer
- License >> Salesforce Platform
- Profile >> Customer
- Make Sure the verified check box is “Unchecked”
- Save

User 4

- Go to Setup >> Administration >> Users >> New User
- Last Name >> Customer2
- Role >> Customer
- License >> Salesforce Platform
- Profile >> Customer
- Make Sure the verified check box is “checked”
- Save



Create An Approval Process For Property Object

- From Setup >> Process Automation >> Approval Process
- Process Name - Property Approval
- Give 2 criteria –
- Location is not equal to blank,
- Verified Equals false.
- Click next and “Next Automated Approver Determined By” Select Manager
- From Record Editability Properties >> Click on Administrators OR the currently assigned approver can edit records during the approval process.
- From Step 5. Select Fields to Display on Approval Page Layout select Property, Owner, Location, Type.
- Click Next and Select the initial Submitters >>
- Owner >> Property Owner
- Roles >> Sales Manager
- Save.
- Add an approval step name “Executive Approval ”
- specify the Criteria >> All record should enter
- click next and select the Approver as “ Sales Executive “ and “Save”
- Add One field Update as “Verified Property”

- Select Object >> Property
- Field to Update >> Verified
- Field Data Type >> CheckBox
- Select CheckBox Option as “True”
- Save
- Add One field Update as “UnVerified Property”
- Select Object >> Property
- Field to Update >> Verified
- Field Data Type >> CheckBox
- Select CheckBox Option as “False”
- Save.
- Activate the Approval Process.

The screenshot displays the Salesforce Lightning Experience interface for managing approval processes. At the top, a navigation bar includes links for Home, Chatter, Contacts, Accounts, Ideas, Reports, and Dashboards. A banner below the navigation bar reads "It's Better in Lightning" with a "Check Readiness" button. The main content area is titled "Approval Processes" and "Property12". A yellow tip box provides instructions for setting up approval processes. Below this, a section titled "Manage Approval Processes For: Property12" shows a list of active approval processes. The table below shows one active process, "Property Approval", with a process order of 1. The "Inactive Approval Processes" section shows "No approval processes available".

Approval Processes
Property12

Approvals are complex business processes that require information gathering and planning before implementing. It is recommended that you follow the instructions below before getting started.

1. Read the [help topic](#)
2. View the [checklist](#)
3. Create a [custom user hierarchical relationship field](#)
4. Create [email templates](#)
5. Create an approval process using either the Jump Start or Standard Wizard
6. Add Approval History Related List to all page layouts
7. Activate the process to deploy to your users

Manage Approval Processes For: **Property12**

A listing of both active and inactive approval processes for **Properties** is displayed below. To create a new approval process, click **Create New Approval Process** then select **Use Jump Start Wizard** to set up your approval process in a few short steps. Or, select **Use Standard Wizard** to configure all approval options.

[Create New Approval Process](#)

Active Approval Processes

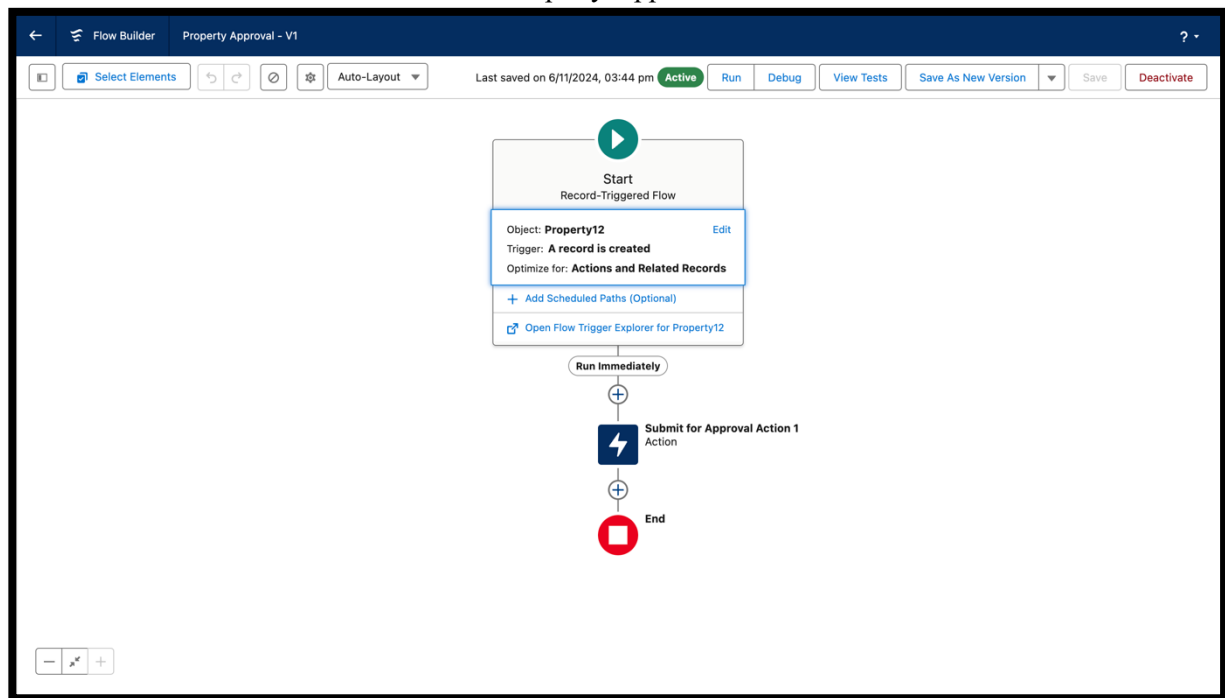
Action	Process Order	Approval Process Name	Description
Edit Deactivate	1	Property Approval	

Inactive Approval Processes

No approval processes available

Create a Record Trigger Flow To Submit The Approval process automatically

- From Setup >> Search for Flows >> Click On New and Select “Record Trigger Flow”.
- Select Object >> Property
- Select “Trigger the flow when” >> “A record is created”
- Set Entry Conditions >> “None”
- Add a “Action” >> “Submit for Approval”
- Give Label >> Approval for property
- Record Id >> {!\$Record.Id}
- Done
- Save the Flow and Give label as “Property Approval” and “Activate”



Create An App Page

- From Setup >> Go to Lightning App Builder >> Click on New >> Select App Page and Click on Next.
- Give Label as “Search your Property” click “Next”.
- Click “header and Left Sidebar” and Click on “Done”
- Click on “Save ” and then click on “Activate”.
- From Page Setting select page activation as “Activate for all Users”.
- From Lightning Experience Click on “Property Details” and click on Add Page“.
- Then Click on “Save”

Lightning App Builder

The Lightning App Builder provides an easy to use graphical interface for creating custom Lightning pages for Salesforce Lightning Experience and mobile app. Lightning pages are built using Lightning components—compact, configurable, and reusable elements that you can drag and drop into regions of the page in the Lightning App Builder.

View: All [Create New View](#)

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z Other All

Action	Label ↑	Name	Namespace Prefix	Description	Type	Created By	Last Modified By
Edit Clone Del	Search your Property	Search_your_Property			App Page	SA, 06/11/2024, 3:46 pm	SA, 06/11/2024, 3:46 pm

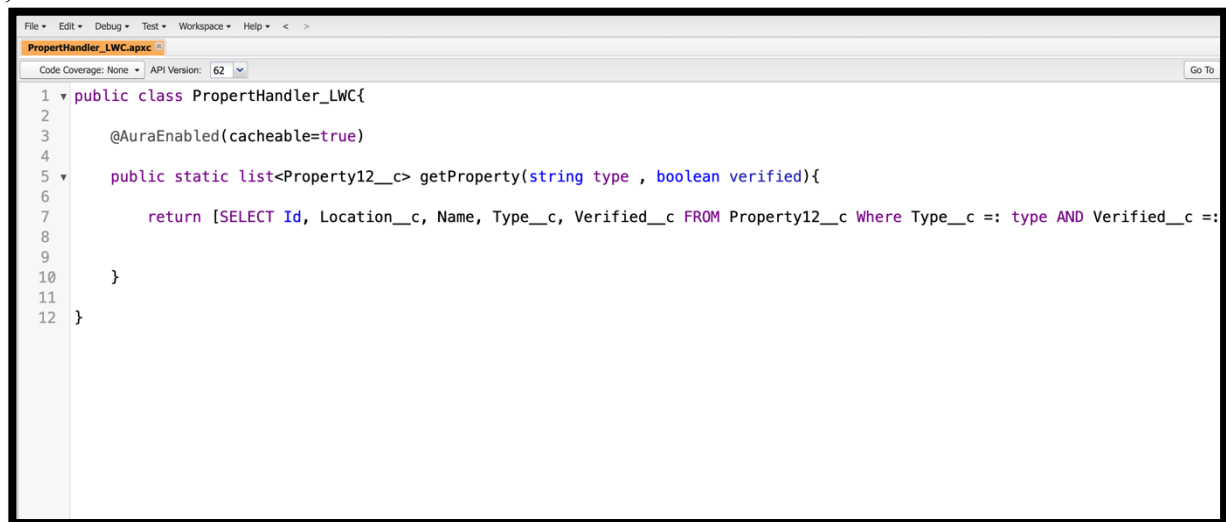
Create A LWC Component

- Create an Apex Class and make it aura enabled and name it “PropertHandler_LWC”

Code: -

```
public class PropertHandler_LWC{

    @AuraEnabled(cacheable=true)
    public static list<Property__c> getProperty(string type , boolean verified){
        return [SELECT Id, Location__c, Property_Name__c, Type__c, Verified__c FROM Property__c
        Where Type__c =: type AND Verified__c =: verified];
    }
}
```



- Create a Lightning Web Component in your VsCode, and (ctrl+shift +P) and click on authorize an org.
- Enter your login id and password to authorize your org.
- Now (ctrl+shift +P) and Create a lightning Web Component and Name it Anything you want to. (Example -)
- In your Html File Write this code :-

Code :-

```
<template>
<lightning-card>
<div class="slds-box">
<div class="slds-text-align_left">
<h1 style="font-size: 20px;"><b>Properties</b></h1>
</div>
<div>
<div class="slds-grid slds-gutters">
<div class="slds-col slds-size_5-of-6">
<lightning-combobox name="Type" label="Property Type" value={typevar}
placeholder="Select Property type"
options={propetyoptions} onchange={changehandler}></lightning-combobox>
</div>
<div class="slds-col slds-size_1-of-6">
<br>
```

```

        <lightning-button-icon    variant="neutral"    icon-name="standard:search"    alternative-
text="Search"
        label="Search" onclick={handleClick}></lightning-button-icon>
    </div>
</div>
</div>
</div>
<template if:true={istru}>
    <div class="slds-box">
        <lightning-datatable    key-field="id"    data={propertylist}    columns={columns}></lightning-
datatable>
    </div>
</template>
<template if:false={isfalse}>
    <div class="slds-box">
        <div style="font-size: 15px;"><b>No properties Are Found !!</b></div>
    </div>
</template>
</lightning-card>
</template>

```

- **In Your Js File Write this code :-**

Code :-

```

import { LightningElement, api, track, wire } from 'lwc';
import getProperty from "@salesforce/apex/PropertyHandler_LWC.getProperty";
import { getRecord } from 'lightning/uiRecordApi';
import USER_ID from '@salesforce/user/Id';
export default class C_01_Property_Management extends LightningElement {
    @api recordId
    userId = USER_ID;
    verifiedvar
    typevar
    isfalse = true;
    istru = false;
    @track propertylist = [];
    columns = [
        { label: 'Property Name', fieldName: 'Property_Name__c' },
        { label: 'Property Type', fieldName: 'Type__c' },
        { label: 'Property Location', fieldName: 'Location__c' },
        { label: 'Property link', fieldName: 'Property_link__c' }
    ]
    propertyoptions = [
        { label: "Commercial", value: "Commercial" },
        { label: "Residential", value: "Residential" },
        { label: "rental", value: "rental" }
    ]
    @wire(getRecord, { recordId: "$userId", fields: ['User.Verified__c'] })
    recordFunction({ data, error }) {
        if (data) {
            console.log(data)
            console.log("This is the User Id ---> "+this.userId);
            this.verifiedvar = data.fields.Verified__c.value;
        } else {
            console.error(error)
            console.log('this is error')
        }
    }
}

```

```

    }
}

changehandler(event) {
    console.log(event.target.value);
    this.typevar = event.target.value;
}
handleClick() {

    getProperty({ type: this.typevar, verified: this.verifiedvar })
        .then((result) => {
            this.isfalse = true;
            console.log(result)
            console.log("This is the User id ---> ' + this.userId);
            console.log("This is the verified values ---> ' + this.verifiedvar);
            if (result != null && result.length != 0) {
                this.istrue = true;
                this.propertylist = result;
                console.log(this.verifiedvar);
                console.log(this.typevar)
            } else {
                this.isfalse = false;
                this.istrue = false;
            }
        })
        .catch((error) => {
            console.log(error)
        })
    }
}

```

- In Your metafile give your targets to deploy the component.

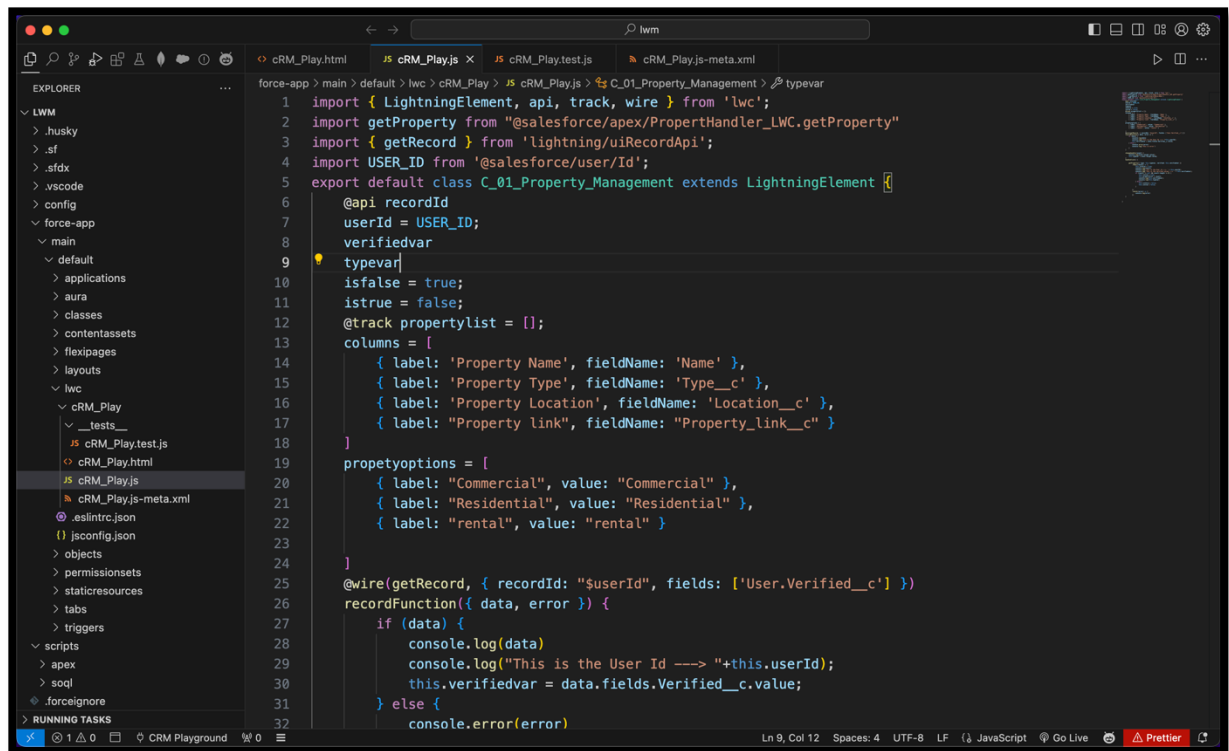
Code :-

```

<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>59.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__RecordPage</target>
        <target>lightning__AppPage</target>
        <target>lightning__HomePage</target>
    </targets>
</LightningComponentBundle>

```

- After Saving all the three Codes , Right Click and deploy this component to the org.



- From Setup >> Go to App Launcher >> Search for Property Details
- On this Page click on gear icon and click on Edit Page
- Drag the Component to your App Page and Save the Page.

Give access on apex classes to profiles

- From Setup >> Search For Apex Classes >> Click on “Security” behind “PropertyHandler_LWC”.
- From Profiles Add “Manager” and “Customer” and “Save”.

